49th IEEE Conference on Decision and Control
December 15-17, 2010
Hilton Atlanta Hotel, Atlanta, GA, USA

The Wireless Control Network: Synthesis and Robustness

Miroslav Pajic, Shreyas Sundaram, Jerome Le Ny, George J. Pappas and Rahul Mangharam

Abstract— **We consider the problem of stabilizing a plant with a network of resource constrained wireless nodes. Traditional networked control schemes are designed with one of the nodes in the network acting as a dedicated controller, while the other nodes simply route information to and from the controller and the plant. We introduce the concept of a Wireless Control Network (WCN) where the entire network itself acts as the controller. Specifically, at each time-step, each node updates its internal state to be a linear combination of the states of the nodes in its neighborhood. We show that this causes the entire network to behave as a linear dynamical system, with sparsity constraints imposed by the network topology. We then provide a numerical design procedure to determine the appropriate linear combinations to be applied by each node so that the transmissions of the nodes closest to the actuators will stabilize the plant. We also show how our design procedure can be modified to maintain mean square stability under packet drops in the network.**

I. Introduction

The advent of low-cost and reliable wireless networks holds great promise for large, spatially distributed industrial control systems. In contrast to the traditional wired interconnections that exist in such systems, wireless networks allow sensor measurements of plant variables to be transmitted to controllers, data centers and plant operators without the need for excessive wiring, thereby yielding gains in efficiency and profitability for the operator.

The topic of control over networks has been intensively studied by researchers over the past decade, leading to design procedures for controllers that are tolerant to network issues such as packet dropouts and transmission delays [1], [2], [3], [4], [5], [6]. These works typically adopt the convention of having a dedicated controller/estimator located somewhere in the network, and study the stability of the closed loop system assuming that the sensor-estimator and/or controller-actuator communication channels are unreliable (dropping packets with a certain probability, for example).

In this paper, we introduce the concept of a *Wireless Control Network* (WCN), which is a paradigm change for control over a wireless network. In a WCN the entire network *itself* acts as a controller, as the computation of the control input is spread over the whole network. We consider a setup where several resource constrained wireless nodes are deployed in the proximity of a plant, with some nodes having access to the sensor measurements (outputs) of the plant, and some nodes placed within the listening range of the plant's actuators. Each node in the WCN is capable of maintaining only a limited internal state. We present a linear iterative strategy for each node to follow, where each node periodically updates its state to be a linear combination of the states of the nodes in its immediate neighborhood. The actuators of the plant also apply linear combinations of the states of the nodes in their neighborhood. Given a linear plant model and the topology of the wireless network, we devise a numerical design procedure that produces the coefficients of the linear combinations for each node and actuator to apply in order to stabilize the plant.

This research has been partially supported by the DARPA MuSyC and the NSF-CNS 0931239 and NSF-MRI Grant 0923518.

The authors are with the Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, PA, USA 19014. Email: {pajic, ssund, jeromel, pappasg, rahulm}@seas.upenn.edu

A. Notation

We use $\mathbf{e}_i$ to denote the i^{th} unit column vector (of appropriate dimension) and the symbol $\mathbf{1}$ denotes the column vector (of appropriate size) consisting of all 1's. The symbol $\mathbf{I}_N$ denotes the $N \times N$ identity matrix. The notation $\text{diag}(\cdot)$ indicates a square matrix with the quantities inside the brackets on the diagonal, and zeros elsewhere. The notation $\text{tr}(\cdot)$ indicates the trace of a square matrix. We will denote the cardinality of a set $\mathcal{S}$ by $|\mathcal{S}|$. The set of nonnegative integers is denoted by $\mathbb{N}$. The notation $\mathbf{A} \succ \mathbf{0} (\succeq \mathbf{0})$ indicates that matrix $\mathbf{A}$ is positive (semi)definite. The set of all $n \times n$ positive definite matrices is denoted by $\mathbb{S}^n_{++}$. A *graph* is denoted by an ordered pair $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V} = \{v_1, v_2, \ldots, v_N\}$ is a set of vertices (or nodes), and $\mathcal{E}$ is a set of ordered pairs of different vertices, called directed edges. The vertices in the set $\mathcal{N}_{v_i} = \{v_j | (v_j, v_i) \in \mathcal{E}\}$ are said to be neighbors of vertex v_i.

II. The Wireless Control Network

Consider the system presented in Fig. 1, where the plant is controlled using a multi-hop, fully synchronized wireless network. In this paper we focus on plants of the form[1]

$$\begin{aligned} \mathbf{x}[k+1] &= \mathbf{A}\mathbf{x}[k] + \mathbf{B}\mathbf{u}[k] \\ \mathbf{y}[k] &= \mathbf{C}\mathbf{x}[k], \end{aligned} \tag{1}$$

with $\mathbf{A} \in \mathbb{R}^{n \times n}, \mathbf{B} \in \mathbb{R}^{n \times m}$ and $\mathbf{C} \in \mathbb{R}^{p \times n}$. The output vector $\mathbf{y}[k] = \begin{bmatrix} y_1[k] & y_2[k] & \ldots & y_p[k] \end{bmatrix}^T$ contains measurements of the plant state vector $\mathbf{x}[k]$ provided by the sensors $s_1, \ldots, s_p$. The input vector $\mathbf{u}[k] = \begin{bmatrix} u_1[k] & u_2[k] & \ldots & u_m[k] \end{bmatrix}^T$ corresponds to the signals applied to the plant by actuators $a_1, \ldots, a_m$.

[1]The plant model can be generalized to include update and measurement noise; if the noise is taken to be independent and identically distributed with a bounded variance, all of our analysis and results will still ensure that the system state is bounded in a mean square sense. For the purposes of clarity, we will therefore omit the noise terms in our discussion.

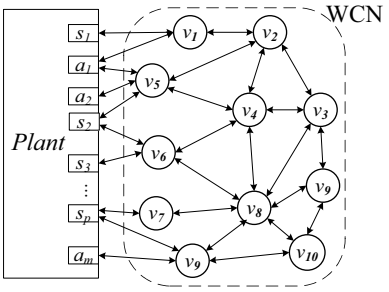

Fig. 1. A multi-hop WCN used as a distributed controller.

The WCN consists of a set of nodes that communicate with each other and with the sensors and actuators installed on the plant. Each node in the network is equipped with a radio transceiver along with (limited) memory and computational capabilities.[2] Similarly, each sensor and actuator on the plant contains a radio transceiver, allowing them to communicate with neighboring nodes. The wireless network is described by a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V} = \{v_1, v_2, \ldots, v_N\}$ is the set of N nodes and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ represents the radio connectivity (communication topology) in the network (i.e., edge $(v_j, v_i) \in \mathcal{E}$ if node v_i can receive information directly from node v_j). We also define $\mathcal{V}_S \subset \mathcal{V}$ as the set of nodes that can receive information directly from at least one sensor, and $\mathcal{V}_A \subset \mathcal{V}$ as the set of nodes whose transmissions can be heard by at least one actuator.

To facilitate our development, we consider a new graph $\bar{\mathcal{G}}$ that includes the plant's sensors and actuators. This graph is obtained by taking the graph $\mathcal{G}$ and adding $p + m$ new vertices $\mathcal{S} \cup \mathcal{A}$, where $\mathcal{S} = \{s_1, s_2, \ldots, s_p\}$ corresponds to the plant's sensors, while $\mathcal{A} = \{a_1, a_2, \ldots, a_m\}$ corresponds to the plant's actuators. Define the edge sets:

$$\mathcal{E}_O = \left\{ (s_l, v_i) \;\middle|\; \begin{array}{c} s_l \in \mathcal{S}, v_i \in \mathcal{V}_S, \\ v_i \text{ can receive values from sensor } s_l \end{array} \right\},$$

$$\mathcal{E}_I = \left\{ (v_i, a_l) \;\middle|\; \begin{array}{c} a_l \in \mathcal{A}, v_i \in \mathcal{V}_A, \\ \text{actuator } a_i \text{ can receive values from } v_i \end{array} \right\}.$$

We then obtain $\bar{\mathcal{G}} = \{\mathcal{V} \cup \mathcal{S} \cup \mathcal{A}, \mathcal{E} \cup \mathcal{E}_I \cup \mathcal{E}_O\}$. Let N_l denote the number of links in $\bar{\mathcal{G}}$ ($N_l = |\mathcal{E} \cup \mathcal{E}_I \cup \mathcal{E}_O|$). We can also define an injective mapping $\Omega : \mathcal{E} \cup \mathcal{E}_I \cup \mathcal{E}_O \to \{1, \ldots, N_l\}$ to enumerate all links in the network. In the rest of the paper, we will sometimes denote a link $(a, b) \in \mathcal{E} \cup \mathcal{E}_I \cup \mathcal{E}_O$ by its label $\Omega(a, b)$ for convenience.

Unlike traditional networked control schemes where a particular node $v_i \in \mathcal{V}$ is designated as the controller (and all other nodes are used to route information between v_i and the plant), the WCN employs a fully distributed control scheme where the entire network itself acts as a controller. At each time-step, every node in the WCN updates its value to be a linear combination of its previous value and the values of its neighbors. In addition, the update procedure of each node from the set $\mathcal{V}_S$ includes a linear combination of the sensor measurements (i.e. plant outputs) from all sensors in its neighborhood. If we let $z_i[k]$ denote node v_i's (scalar) state at time step k, we obtain the update procedure:[3]

$$z_i[k+1] = w_{ii} z_i[k] + \sum_{v_j \in \mathcal{N}_{v_i}} w_{ij} z_j[k] + \sum_{s_j \in \mathcal{N}_{v_i}} h_{ij} y_j[k]. \quad (2)$$

Each plant input $u_i[k], i \in \{1, \ldots, m\}$ is taken to be a linear combination of values from the nodes in the neighborhood of the actuator a_i:

$$u_i[k] = \sum_{j \in \mathcal{N}_{a_i}} g_{ij} z_j[k]. \quad (3)$$

The scalars w_{ij}, h_{ij} and g_{ij} specify the linear combinations that are computed by each node and actuator in the network. If we aggregate the values of all nodes at time step k into the value vector $\mathbf{z}[k] = \begin{bmatrix} z_1[k] & z_2[k] & \cdots & z_N[k] \end{bmatrix}^T$, the linear iterative procedure for the entire system can be described as:

$$\begin{aligned} \mathbf{z}[k+1] &= \mathbf{W}\mathbf{z}[k] + \mathbf{H}\mathbf{y}[k] \;, \\ \mathbf{u}[k] &= \mathbf{G}\mathbf{z}[k] \end{aligned}$$

for all $k \in \mathbb{N}$ ($\mathbf{W} \in \mathbb{R}^{N \times N}, \mathbf{H} \in \mathbb{R}^{N \times p}, \mathbf{G} \in \mathbb{R}^{m \times N}$). In the above equation, for all $i \in \{1, \ldots, N\}$, $w_{ij} = 0$ if $v_j \notin \mathcal{N}_{v_i} \cup \{v_i\}$, $h_{ij} = 0$ if $s_j \notin \mathcal{N}_{v_i}$, and $g_{ij} = 0$ if $v_j \notin \mathcal{N}_{a_i}$. Thus the matrices $\mathbf{W}, \mathbf{H}$ and $\mathbf{G}$ are *structured*, meaning that they have sparsity constraints determined by the topology of the WCN. Throughout the rest of the paper, we will define Ψ to be the set of all tuples $(\mathbf{W}, \mathbf{H}, \mathbf{G}) \in \mathbb{R}^{N \times N} \times \mathbb{R}^{N \times p} \times \mathbb{R}^{m \times N}$ satisfying the aforementioned sparsity constraints. If we denote the overall system state by $\hat{\mathbf{x}}[k] = [\mathbf{x}[k]^T \; \mathbf{z}[k]^T]^T$, the closed-loop system evolves as:

$$\hat{\mathbf{x}}[k+1] = \begin{bmatrix} \mathbf{A} & \mathbf{B}\mathbf{G} \\ \mathbf{H}\mathbf{C} & \mathbf{W} \end{bmatrix} \begin{bmatrix} \mathbf{x}[k] \\ \mathbf{z}[k] \end{bmatrix} \triangleq \hat{\mathbf{A}}\hat{\mathbf{x}}[k]. \quad (4)$$

In this paper we describe an algorithm that can be used to find an element of Ψ that causes the matrix $\hat{\mathbf{A}}$ to be Schur (provided such an element exists).[4] We first consider the case with reliable communication links, and then extend our approach to accommodate independent Bernoulli link failures in the network. In the companion paper [8], we also show how to design an "Intrusion Detection System" for this control scheme, which observes the transmissions of certain nodes in order to identify any abnormal behavior.

A. Advantages of Wireless Control Networks

In the context of multi-hop embedded wireless networks used for control, the WCN has the following advantages.

1. Low overhead: The proposed scheme is computationally inexpensive since a node only needs to compute a linear combination of its value and values of its neighbors. Thus, the WCN can be easily implemented even on resource constrained, low-power wireless nodes (e.g., FireFly, tMote, micaZ) using very simple, periodic tasks executed on a real-time operating system (e.g., nano-RK [9] or TinyOS [10]). Also, as each node is only required to transmit its state once per frame, the proposed scheme can be easily "piggy-backed" into existing wireless networks that assign a transmission

[2] We will model these resource constraints by limiting the size of the state vector maintained by each node. To present our results, we will focus on the case where each node's state is represented as a scalar. Our results can be readily extended to the more general case where each node can maintain a vector state with possibly different dimensions. For details see [7].

[3] The neighborhood $\mathcal{N}_v$ of a vertex v is with respect to the graph $\bar{\mathcal{G}}$.

[4] A matrix is Schur if all of its eigenvalues are inside the unit circle.

slot for each node to maintain network related information (e.g., wireless systems for factory automation based on the ISA100.11a standard [11] or wirelessHART [12]). This also allows the possibility of using the proposed scheme as a backup mechanism in traditional networked control systems. Specifically, if the primary control mechanism (i.e., dedicated controller) in the existing networked control infrastructure fails, the wireless network itself can take over the role of stabilizing the plant until the primary controller is restored.

2. Compositionality: The WCN allows *compositionality*, meaning that an existing design can be easily extended to control new subsystems that are added to the plant. In subsequent sections we describe how the WCN can be used to stabilize a given plant (or set of interconnected plants). However, suppose that some new subsystems (or plants) are added in the proximity of the WCN. Rather than recalculating a stabilizing set of linear combinations that would work for all plants simultaneously, one can instead calculate a separate stabilizing set of linear combinations for each of the new plants, with corresponding separate states maintained by each node. If P is the total number of different plants, each node would calculate P linear combinations of the values received from its neighbors and group the P corresponding states in one transmission per frame.[5] This enables a completely decoupled computation of the matrices $\{\mathbf{W}_i, \mathbf{H}_i, \mathbf{G}_i\}_{i=1}^{P}$ that guarantee MSS for each of the P plants, although physically realized by the same WCN.

3. Simple scheduling: The presented scheme does not require complex communication scheduling, since each node needs to transmit exactly once in a frame and the WCN does not impose end-to-end delay constraints (i.e., nodes close to the actuators do not need to wait for information to propagate all the way from the sensors). The only requirement of the communication schedule is to be conflict-free (i.e., two transmission scheduled at the same slot should not affect each other). Thus, if d_i is the maximal degree of the interference graph,[6] a *static* conflict-free schedule can be derived using graph coloring, with at most d_i slots in a frame. Since the duration of a frame is equal to the plant's sampling period, the minimal sampling period of the plant is equal to $d_i T_{slot}$, where T_{slot} is the duration of a communication slot. In contrast, some techniques used for traditional networked control systems impose a requirement that the sampling period be greater than the end-to-end delay, causing the minimal sampling period to directly depend on the network diameter.

4. Multiple sensing/actuation points: The WCN can handle plants with multiple geographically distributed sensors and actuators, a case that is not easily handled by the "sensor $\rightarrow$ channel $\rightarrow$ controller/estimator $\rightarrow$ channel $\rightarrow$ actuator" setup that is commonly adopted in networked control design.

[5]This is usually not a limitation even for low-bandwidth 802.15.4 networks (where each transmission packet can contain up to 1024 bits). For instance, if a node maintains a scalar 32 bit state for each plant, then up to 32 plants can be controlled in parallel.

[6]The interference graph is defined as $\bar{\mathcal{G}}_{Int} = \{\mathcal{V} \cup \mathcal{S} \cup \mathcal{A}, \mathcal{E}_{Int}\}$, where a link between two nodes (or a node and a sensor/actuator) indicates that they can interfere with each other (i.e., cannot transmit simultaneously).

III. Stabilizing The Closed-loop System

From Eq. (4), the closed-loop system is stable if the matrices $\mathbf{A}, \mathbf{W}, \mathbf{H}$ are chosen so that $\hat{\mathbf{A}}$ is Schur. The traditional approach to achieving this would be to find a positive definite matrix $\mathbf{X}$ satisfying the Lyapunov inequality $\mathbf{X} - \hat{\mathbf{A}}^T \mathbf{X} \hat{\mathbf{A}} \succ 0$, or equivalently, $\left[\begin{smallmatrix} \mathbf{X} & \hat{\mathbf{A}}^T\mathbf{X} \\ \mathbf{X}\hat{\mathbf{A}} & \mathbf{X} \end{smallmatrix}\right] \succ 0$. The condition is not linear in the design parameters $\mathbf{X}, \mathbf{W}, \mathbf{H}, \mathbf{G}$; this is of no consequence in standard controller design (without structural constraints on the design matrices), because this condition can be converted to a LMI via an appropriate transformation of the system matrices (e.g., as done in [13]). However, the fact that the matrices are *structured* in our framework prevents us from directly applying these standard procedures. Still, the following alternative characterization of stability of structured systems from [14] offers a solution.

Theorem 1: ([14]) A matrix $\hat{\mathbf{A}}$ is Schur iff there exist symmetric, positive-definite matrices $\mathbf{X}$ and $\mathbf{Y}$ such that $\left[\begin{smallmatrix} \mathbf{X} & \hat{\mathbf{A}}^T \\ \hat{\mathbf{A}} & \mathbf{Y} \end{smallmatrix}\right] \succ 0$, $\mathbf{X} = \mathbf{Y}^{-1}$. □

The theorem provides a matrix inequality that is *linear* in the design variables $\mathbf{W}, \mathbf{G}$ and $\mathbf{H}$, but suffers from the fact that the constraint $\mathbf{X} = \mathbf{Y}^{-1}$ is nonconvex. However, as pointed out in [14], constraints of this form commonly occur in the design of static output feedback controllers, and there are various numerical methods to address this issue. One particularly appealing approach, suggested in [15], [16], is to approximate the constraint $\mathbf{X} = \mathbf{Y}^{-1}$ with a linear optimization problem using the following lemma (the proof can be obtained in [7]).

Lemma 1: Positive-definite matrices $\mathbf{X}, \mathbf{Y}$ satisfy the constraint $\mathbf{X} = \mathbf{Y}^{-1}$ iff they are optimal points for the problem

$$(P): \quad \min tr(\mathbf{XY}), \;\; s.t.\; \mathbf{X} \succeq \mathbf{Y}^{-1}, \;\; \mathbf{X}, \mathbf{Y} \in \mathbb{S}^n_{++}$$

and the optimal cost of the problem is n.

Using the Schur complement, the constraint $\mathbf{X} \succeq \mathbf{Y}^{-1}$ in the above lemma can be readily transformed to the form $\left[\begin{smallmatrix} \mathbf{X} & \mathbf{I} \\ \mathbf{I} & \mathbf{Y} \end{smallmatrix}\right] \succeq 0$. From Eq. (4), Theorem 1 and Lemma 1 the following corollary can be obtained.

Corollary 1: The WCN can stabilize the system if and only if the following optimization problem

$$\min tr(\mathbf{XY}), \tag{5}$$

$$\left[\begin{array}{cc} \mathbf{X} & \hat{\mathbf{A}}^T \\ \hat{\mathbf{A}} & \mathbf{Y} \end{array}\right] \succ \mathbf{0}, \;\; \left[\begin{array}{cc} \mathbf{X} & \mathbf{I} \\ \mathbf{I} & \mathbf{Y} \end{array}\right] \succeq \mathbf{0}, \tag{6}$$

$$\hat{\mathbf{A}} = \left[\begin{array}{cc} \mathbf{A} & \mathbf{BG} \\ \mathbf{HC} & \mathbf{W} \end{array}\right], \tag{7}$$

$$(\mathbf{W}, \mathbf{H}, \mathbf{G}) \in \Psi, \;\; \mathbf{X}, \mathbf{Y} \in \mathbb{S}^{n+N}_{++} \tag{8}$$

is feasible with optimal cost $n + N$.

Note that with the exception of the objective function (5), all of the constraints in the above corollary are linear in the unknown parameters, and can readily be solved using LMI tools. In [16], El Ghaoui *et al.* showed that the nonconvex function tr$(\mathbf{XY})$ can be replaced with a linear approximation

$$\phi_{lin}(\mathbf{X}, \mathbf{Y}) = constant + tr(\mathbf{Y_0 X} + \mathbf{X_0 Y}),$$

for any given matrices $\mathbf{X}_0$ and $\mathbf{Y}_0$. With this insight, [15], [16] showed that an iterative algorithm can be used to minimize $\text{tr}(\mathbf{XY})$, while ensuring satisfaction of LMI constraints. For our application, the iterative approach proposed in those papers can be formulated as Algorithm 1.

Algorithm 1 Stabilizing closed-loop system with the WCN

1. Find feasible points $\mathbf{X_0}, \mathbf{Y_0}, \mathbf{W_0}, \mathbf{H_0}, \mathbf{G_0}$ that satisfy the constraints (6)-(8). If a feasible point does not exist, then it is not possible to stabilize the system with this network topology.

2. At iteration k ($k \geq 0$), from $\mathbf{X_k}, \mathbf{Y_k}$ obtain the matrices $\mathbf{X_{k+1}}, \mathbf{Y_{k+1}}, \mathbf{W_{k+1}}, \mathbf{H_{k+1}}, \mathbf{G_{k+1}}$ by solving the following LMI problem

$$\min tr(\mathbf{Y_k X_{k+1}} + \mathbf{X_k Y_{k+1}})$$
$$\begin{bmatrix} \mathbf{X}_{k+1} & \hat{\mathbf{A}}_{k+1}^T \\ \hat{\mathbf{A}}_{k+1} & \mathbf{Y}_{k+1} \end{bmatrix} \succ 0, \quad \begin{bmatrix} \mathbf{X_{k+1}} & \mathbf{I} \\ \mathbf{I} & \mathbf{Y_{k+1}} \end{bmatrix} \succeq 0,$$
$$\hat{\mathbf{A}}_{k+1} = \begin{bmatrix} \mathbf{A} & \mathbf{B G}_{k+1} \\ \mathbf{H}_{k+1}\mathbf{C} & \mathbf{W_{k+1}} \end{bmatrix},$$
$$(\mathbf{W_{k+1}}, \mathbf{H}_{k+1}, \mathbf{G}_{k+1}) \in \Psi, \;\; \mathbf{X}_{k+1}, \mathbf{Y}_{k+1} \in \mathbb{S}_{++}^{n+N}.$$

3. If the matrix

$$\hat{\mathbf{A}}_{k+1} = \begin{bmatrix} \mathbf{A} & \mathbf{B G}_{k+1} \\ \mathbf{H}_{k+1}\mathbf{C} & \mathbf{W}_{k+1} \end{bmatrix}$$

is Schur, stop the algorithm. Otherwise, set $k = k+1$ and go to the step **2**.

In [16] the authors showed that the sequence $t_k = tr(\mathbf{Y}_k\mathbf{X}_{k+1} + \mathbf{X}_k\mathbf{Y}_{k+1})$ always converges. In addition, if it converges to $2(n+N)$ the condition $\mathbf{Y} = \mathbf{X}^{-1}$ can be satisfied under the given LMI constraints. A similar proof can be constructed in this case, leading to the following theorem.

Theorem 2: **Algorithm** 1 determines a tuple $(\mathbf{W}, \mathbf{H}, \mathbf{G}) \in \Psi$ that causes the matrix $\hat{\mathbf{A}}(\mathbf{W}, \mathbf{H}, \mathbf{G})$ to be Schur if the sequence t_k converges to $2(n+N)$. □

While each iteration of the above algorithm is a convex optimization problem (which can be efficiently solved using standard LMI toolboxes), we do not have a characterization of the number of iterations required for the algorithm to converge.

IV. Stabilization Despite Unreliable Communication Links

In this section we focus on more "realistic" system models, where potential message drops are taken into consideration. In this case the system's evolution can be described as $\hat{\mathbf{x}}[k+1] = \begin{bmatrix} \mathbf{A} & \mathbf{B G}_{\theta(\mathbf{k})} \\ \mathbf{H}_{\theta(\mathbf{k})}\mathbf{C} & \mathbf{W}_{\theta(\mathbf{k})} \end{bmatrix} \hat{\mathbf{x}}[k]$, where $\hat{\mathbf{x}}[k] \in \mathbb{R}^{n+N}$ is the overall system's state and the subscript $\theta(k)$ describes time-variations in the matrices $(\mathbf{W}, \mathbf{H}, \mathbf{G})$ caused by (probabilistic) drops of communication packets. The focus of this section is a design procedure that can guarantee mean-square stability (MSS) of the closed loop system, defined as:

Definition 1: ([17]) The system is mean-square stable if for any initial state $\hat{\mathbf{x}}[0]$, $\lim_{k\to\infty} \mathbb{E}\left[\|\hat{\mathbf{x}}[k]\|^2\right] = 0$, where the expectation is with respect to the probability distribution of the packet drop sequence $\theta(k)$. □

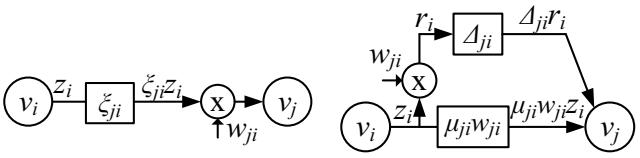

Fig. 2. Remote control over fading channel; (a) A link between nodes v_i and v_j; (b) Link transformation into a robust control form.

There are relatively few results that explicitly consider packet drops in networked control systems with general topologies. The paper [5] considered the problem of the optimal location for a controller in a network, and the papers [6], [18] considered the issue of allowing intermediate nodes to encode information that they are routing to the controller. All of these papers assume a single sensor and actuation point on the plant, consider the existence of a designated controller within the network, and focus on the issue of transmitting the sensor measurements (or some function of them) to that controller.

The topic of modeling networks with unreliable channels was also considered in [17], where it was shown that such networks can be cast in a robust control framework. In the framework of [17], a communication link is modeled over time as a memoryless, discrete, independent and identically distributed (IID) random process ξ,[7] which maps each transmitted value $t_x[k]$ into a received value $r_x[k] = \xi[k]t_x[k]$.[8] For arbitrary nodes v_i and v_j consider a communication link $(v_i, v_j) \in \mathcal{E}$ with weight w_{ji} (as shown in Fig. 2(a)). In the rest of the paper we will also denote this link as $t = \Omega(v_i, v_j)$ and its weight as w_t, h_t or g_t. In addition, all variables related to the link will be denoted with index t (e.g. $\xi_t[k]$, instead of $\xi_{ji}[k]$). The contribution of the node v_i to the linear combination calculated by node v_j at time k can be represented as $w_t\xi_t[k]z_i[k]$ where ξ_t has mean $\mu_t = \mathbf{E}[\xi_t[k]]$ and a finite variance $\sigma_t^2 = \mathbf{E}\left[(\xi_t[k] - \mu_t)^2\right]$.

Following the approach in [17], we consider the link transformation shown in Fig. 2(b). By writing $\xi_t[k] = \mu_t + \Delta_t[k]$, where $\Delta_t[k]$ is a zero-mean random variable with variance σ_t^2, the original unreliable link is modeled as a combination of the deterministic link (without message drops) with gain μ_t and the random link described with gain $\Delta_t[k]$. Let $r_t[k]$ denote the signal that is injected into the t^{th} link, scaled by the weight on that link:

$$r_t[k] = \begin{cases} h_t y_i[k] & \text{if } t = \Omega(s_i, v_j), \\ w_t z_i[k] & \text{if } t = \Omega(v_i, v_j), \\ g_t z_i[k] & \text{if } t = \Omega(v_i, a_j). \end{cases}$$

Stacking all of the $r_t[k]$'s in a vector $\mathbf{r}[k]$ of length N_l, we can write

$$\mathbf{r}[k] = \mathbf{J}^{or}\begin{bmatrix} \mathbf{y}[k] \\ \mathbf{z}[k] \end{bmatrix} = \underbrace{\mathbf{J}^{or}\begin{bmatrix} \mathbf{C} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_N \end{bmatrix}}_{\tilde{\mathbf{J}}^{or}} \hat{\mathbf{x}}[k], \qquad (9)$$

[7]Here IID implies that the random variables $\{\xi[k]\}_{k\geq 0}$ are IID.

[8]Note that a Bernoulli packet drop channel can be modeled by setting $\xi[k] = 0$ with probability p and 1 with probability $1-p$.

where each row of the matrix $\mathbf{J}^{or} \in \mathbb{R}^{N_l \times (N+p)}$ contains a single nonzero element, equal to a gain w_t, h_t or g_t.

Based on the link transformation shown in Fig. 2(b), and using (2), the update equation for each node v_j is

$$z_j[k+1] = w_{jj} z_j[k] + \sum_{t=\Omega(v_i,v_j)} \mu_t w_t z_i[k] + \sum_{t=\Omega(s_i,v_j)} \mu_t h_t y_i[k] + \sum_{t=\Omega(v_i,v_j)} \Delta_t[k] r_t[k] + \sum_{t=\Omega(s_i,v_j)} \Delta_t[k] r_t[k].$$

Also, from (3), the input value applied by each actuator at time step k is

$$u_j[k] = \sum_{t=\Omega(v_i,a_j)} \mu_t g_t z_i[k] + \sum_{t=\Omega(v_i,a_j)} \Delta_t[k] r_t[k].$$

Let $\mathbf{\Delta}[k] = \text{diag}(\{\Delta_t[k]\}_{t=1}^{N_l})$, so that the above expressions can be written in vector form as

$$\mathbf{z}[k+1] = \mathbf{W}_\mu \mathbf{z}[k] + \mathbf{H}_\mu \mathbf{y}[k] + \mathbf{J}_v^{dst} \mathbf{\Delta}[k] \mathbf{r}[k],$$
$$\mathbf{u}[k] = \mathbf{G}_\mu \mathbf{z}[k] + \mathbf{J}_u^{dst} \mathbf{\Delta}[k] \mathbf{r}[k],$$

where each nonzero entry of matrices $\mathbf{W}_\mu, \mathbf{H}_\mu$ and $\mathbf{G}_\mu$ (except the diagonal entries of $\mathbf{W}_\mu$) is of the form $\mu_t w_t, \mu_t h_t$ and $\mu_t g_t$, respectively. Each entry in the matrices $\mathbf{J}_v^{dst}$ and $\mathbf{J}_u^{dst}$ is either 0 or 1. Specifically, each row of those matrices simply selects which elements of the vector $\mathbf{\Delta}[k]\mathbf{r}[k]$ are added to the linear combinations calculated by the actuators and the wireless nodes. From this, the overall system (with potential message drops) can be represented as:

$$\hat{\mathbf{x}}[k+1] = \underbrace{\begin{bmatrix} \mathbf{A} & \mathbf{B}\mathbf{G}_\mu \\ \mathbf{H}_\mu \mathbf{C} & \mathbf{W}_\mu \end{bmatrix}}_{\hat{\mathbf{A}}_\mu} \hat{\mathbf{x}}[k] + \underbrace{\begin{bmatrix} \mathbf{B} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_N \end{bmatrix} \mathbf{J}^{dst}}_{\hat{\mathbf{J}}^{dst}} \mathbf{\Delta}[k] \mathbf{r}[k], \tag{10}$$

with $\mathbf{J}^{dst} = \begin{bmatrix} \mathbf{J}_u^{dst} \\ \mathbf{J}_v^{dst} \end{bmatrix} \in \mathbb{R}^{(m+N) \times N_l}$ and $\mathbf{r}[k]$ given by (9).

As previously mentioned, an assumption is made that $\Delta_t[1], \Delta_t[2], \ldots \Delta_t[k], \ldots$ are independent zero-mean random variables with variance σ_t^2. In addition, we assume that all random variables, $\Delta_1, \ldots, \Delta_{N_l}$ are independent. With this assumption, using the approach from [17], we obtain the following result (for a detailed proof see [7]):

Theorem 3: The system from Eq. (10) is MSS if and only if there exists a positive-definite matrix $\mathbf{X}$ and scalars $\alpha_1, \ldots, \alpha_{N_l}$ satisfying the LMIs

$$\begin{aligned} &\mathbf{X} \succ \hat{\mathbf{A}}_\mu \mathbf{X} \hat{\mathbf{A}}_\mu^T + \hat{\mathbf{J}}^{dst} \text{diag}\{\alpha\} (\hat{\mathbf{J}}^{dst})^T \\ &\alpha_i \geq \sigma_i^2 (\hat{\mathbf{J}}^{or})_i \mathbf{X} (\hat{\mathbf{J}}^{or})_i^T, \ \forall i \in \{1, \ldots, N_l\} \end{aligned} \tag{11}$$

where $(\hat{\mathbf{J}}^{or})_i$ denotes the i^{th} row of the matrix $\hat{\mathbf{J}}^{or}$. □

As in Section III, **Algorithm** 2 can be constructed to solve the inequalities presented in the above theorem (since the matrix $\mathbf{J}^{dst}$ and σ_i's are constants).

Theorem 4: **Algorithm** 2 will determine the tuple $(\mathbf{W}, \mathbf{G}, \mathbf{H}) \in \Psi$ that guarantees MSS of the system under the given links' failure distribution if the sequence $t_k = tr(\mathbf{Y}_k \mathbf{X}_{k+1} + \mathbf{X}_k \mathbf{Y}_{k+1})$ converges to $2(n+N)$. □

Remark 1: If we consider all links to have the same $\sigma_i = \sigma$, the largest value of packet loss for which the system is MSS can be found by allowing $\sigma \in \mathbb{R}$ to be a variable. This causes the last matrix inequality in step 2 of Algorithm 2 to be a bilinear constraint, but this can be handled by using bisection on the parameter $\sigma \in \mathbb{R}$ (e.g., as done in [13]). □

Algorithm 2 Stabilizing the closed-loop system with unreliable communication links using the WCN

1. Find feasible points $\mathbf{X_0}, \mathbf{Y_0}, \mathbf{W_0}, \mathbf{H_0}, \mathbf{G_0}$ that satisfy the constraints (11), where $\mathbf{X_0}, \mathbf{Y_0} \in \mathbb{S}_{++}^{n+N}$ and:

$$\begin{bmatrix} \mathbf{X_0} & \mathbf{I} \\ \mathbf{I} & \mathbf{Y_0} \end{bmatrix} \succeq 0, \ (\mathbf{W_0}, \mathbf{H_0}, \mathbf{G_0}) \in \Psi$$

If there is no feasible point, it is not possible to obtain MSS with this network topology and distribution on the communication links.

2. At iteration k, $(k \geq 0)$ from $\mathbf{X_k}, \mathbf{Y_k}$ obtain the matrices $\mathbf{X_{k+1}}, \mathbf{Y_{k+1}}, \mathbf{W}_{\mu,\mathbf{k+1}}, \mathbf{H}_{\mu,\mathbf{k+1}}, \mathbf{G}_{\mu,\mathbf{k+1}}$ and a vector $\alpha_{k+1} \in \mathbb{R}^{N_l}$ by solving the following LMI problem

$$\min tr(\mathbf{Y_k X_{k+1}} + \mathbf{X_k Y_{k+1}})$$
$$\begin{bmatrix} \mathbf{X}_{k+1} - (\hat{\mathbf{J}}^{dst}) diag\{\alpha_{k+1}\} (\hat{\mathbf{J}}^{dst})^T & \hat{\mathbf{A}}_{\mu,k+1} \\ \hat{\mathbf{A}}_{\mu,k+1}^T & \mathbf{Y}_{k+1} \end{bmatrix} \succ 0,$$
$$\begin{bmatrix} \mathbf{X_{k+1}} & \mathbf{I} \\ \mathbf{I} & \mathbf{Y_{k+1}} \end{bmatrix} \succeq 0,$$
$$\hat{\mathbf{A}}_{\mu,k+1} = \begin{bmatrix} \mathbf{A} & \mathbf{B}\mathbf{G}_{\mu,\mathbf{k+1}} \\ \mathbf{H}_{\mu,\mathbf{k+1}}\mathbf{C} & \mathbf{W}_{\mu,\mathbf{k+1}} \end{bmatrix},$$
$$\begin{bmatrix} \alpha_{i,k+1} & \sigma_i(\hat{\mathbf{J}}_{k+1}^{or})_i \\ \sigma_i(\hat{\mathbf{J}}_{k+1}^{or})_i^T & \mathbf{Y}_{k+1} \end{bmatrix} \succeq 0, \ 1 \leq i \leq N_l,$$
$$(\mathbf{W}_{\mu,\mathbf{k+1}}, \mathbf{H}_{\mu,k+1}, \mathbf{G}_{\mu,k+1}) \in \Psi, \ \mathbf{X}_{k+1}, \mathbf{Y}_{k+1} \in \mathbb{S}_{++}^{n+N}$$

3. Stop the algorithm if the following conditions are true

$$\mathbf{X}_{k+1} \succ \hat{\mathbf{A}}_{\mu,k+1} \mathbf{X}_{k+1} \hat{\mathbf{A}}_{\mu,k+1}^T + \hat{\mathbf{J}}^{dst} diag\{\alpha_{k+1}\} (\hat{\mathbf{J}}^{dst})^T$$
$$\alpha_{i,k+1} \geq \sigma_i^2 (\mathbf{J}_{k+1}^{or})_i \mathbf{X}_{k+1} (\mathbf{J}_{k+1}^{or})_i^T, \ 1 \leq i \leq N_l.$$

Otherwise, set $k = k+1$ and go to step **2**.

V. Discussion and Example

At first glance, the control scheme that we have presented in this paper might seem to introduce some delay into the feedback loop (since the sensor nodes and actuator nodes might be separated by multiple intermediate nodes, each taking one time-step to propagate information), which might limit the class of plants that can be stabilized with this method. However, the relationship between the WCN and the traditional notions of delay introduced by the feedback loop is not as obvious as it might appear at first glance. Specifically, note that we allow each node in the network to maintain a value that is a function of its previous value and the values of all its neighbors, rather than simply routing values to a controller. This simple modification causes the network to essentially act as a *linear dynamical system* with sparsity constraints in the system matrices; in other words, this control scheme should be viewed as a dynamic compensator, rather than a static feedback gain at the end of a chain of delay elements. The following example shows that this fact allows our scheme to stabilize plants that cannot be stabilized with delayed static feedback.

Consider the single-state plant shown in Fig. 3 (with $\alpha >$

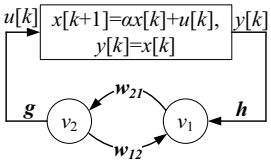

Fig. 3. An example of a WCN.

1), which is to be controlled by a network with two nodes v_1 and v_2. Node v_1 receives the plant output $y[k] = x[k]$ at each time-step k, and the input to the plant is taken to be a scaled version of the transmission of the node v_2 (i.e., $u[k] = gz_2[k]$, for some scalar g). If the nodes apply the linear strategy that we study in this paper, the closed loop system evolves according to

$$\begin{bmatrix} x[k+1] \\ z_1[k+1] \\ z_2[k+1] \end{bmatrix} = \begin{bmatrix} \alpha & 0 & g \\ h & w_{11} & w_{12} \\ 0 & w_{21} & w_{22} \end{bmatrix} \begin{bmatrix} x[k] \\ z_1[k] \\ z_2[k] \end{bmatrix} , \qquad (12)$$

for some scalars $w_{11}, w_{12}, w_{21}, w_{22}, g$ and h. One can verify that these scalars can be chosen so that the closed loop system is stable, regardless of the value of α. For example, if one chooses the values $g = h = 1$, $w_{11} = 0$, $w_{12} = \frac{1}{\alpha}$, $w_{21} = -\alpha^3$ and $w_{22} = -\alpha$, the closed-loop system will have all poles at zero.

Now, consider a control scheme where node v_1 simply forwards the state measurement to v_2 at each time-step, and v_2 sends this value to the actuator where the input $u[k] = gz_2[k]$ is applied. This can be modeled by setting $w_{11} = w_{12} = w_{22} = 0$, $w_{21} = 1$, and $h = 1$ in (12). The characteristic polynomial of this system is $z^2(z-\alpha)-g$, and one can show (e.g., using the root locus) that it is possible to find a g such that this polynomial has all roots inside the unit circle if and only if $|\alpha| < \frac{3}{2}$. In other words, the delay introduced by this routing scheme limits the class of plants that can be stabilized. One obtains stability for arbitrary values of α only by allowing both v_1 and v_2 to update their values with a linear strategy (as demonstrated above).

To illustrate the application of our design procedure[9] from the previous sections, suppose that each link in the network is modeled as an independent Bernoulli process with probability of losing a packet equal to p (the variance of each process is $\sigma^2 = p(1-p)$). Obviously, for $\alpha > 1$ the plant is unstable, even for reliable communication links ($p = 0$). For $\alpha = 2$ and $p = 0.5\%$ Algorithm 2 converges after 51 iterations[10] to the stable configuration

$$\mathbf{W} = \begin{bmatrix} 0.228 & 0.965 \\ -2.872 & -1.660 \end{bmatrix}, \mathbf{H} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \mathbf{G} = \begin{bmatrix} 0 & 1.837 \end{bmatrix}.$$

Using the bisection method described in the Remark 1, we computed that the maximal probability of message drops, p_{max}, for which there exists a tuple $(\mathbf{W}, \mathbf{H}, \mathbf{G}) \in \Psi$ that guarantees MSS is $p_{max} = 0.69\%$. In addition, networks with $N = 3$ and $N = 4$ nodes were considered, where the graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ is complete ($\mathcal{V} = \{v_1, \ldots, v_N\}$). In these cases maximal probabilities are $p_{max} = 0.74\%$ and $p_{max} = 0.77\%$ respectively. As can be seen, adding more nodes in the network increases the robustness of the system to packet drops in the wireless network.

[9]More complex examples with larger plants and networks (e.g. a plant with 30 states controlled by a mesh network with 16 nodes) along with examples where nodes maintain vector states can be found in [7].

[10]The number of iterations needed before the algorithm converges to a stable configuration depends on initial points $\mathbf{X_0}, \mathbf{Y_0}$.

REFERENCES

[1] C. N. Hadjicostis and R. Touri, "Feedback control utilizing packet dropping network links," in *Proc. of the 41st IEEE Conference on Decision and Control*, 2002, pp. 1205–1210.

[2] O. C. Imer, S. Yuksel, and T. Basar, "Optimal control of LTI systems over unreliable communication links," *Automatica*, vol. 42, no. 9, pp. 1429–1439, Sep. 2006.

[3] J. P. Hespanha, P. Naghshtabrizi, and Y. Xu, "A survey of recent results in networked control systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 138–162, Jan. 2007.

[4] L. Schenato, B. Sinopoli, M. Franceschetti, K. Poolla, and S. S. Sastry, "Foundations of control and estimation over lossy networks," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 163–187, 2007.

[5] C. L. Robinson and P. R. Kumar, "Optimizing controller location in networked control systems with packet drops," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 4, pp. 661–671, May 2008.

[6] V. Gupta, A. F. Dana, J. Hespanha, R. M. Murray, and B. Hassibi, "Data transmission over networks for estimation and control," *IEEE Transactions on Automatic Control*, vol. 54, no. 8, pp. 1807–1819, Aug. 2009.

[7] M. Pajic, S. Sundaram, R. Mangharam, and G. J. Pappas, "The Wireless Control Network," University of Pennsylvania, Tech. Rep., Mar. 2010.

[8] S. Sundaram, M. Pajic, C. Hadjicostis, R. Mangharam, and G. J. Pappas, "The Wireless Control Network: Monitoring for Malicious Behavior," in *Proceedings of the 49th IEEE Conference on Decision and Control*, 2010.

[9] "The nano-RK Sensor Real-Time Operating System," http://nanork.org.

[10] P. Levis, S. Madden, J. Polastre, R. Szewczyk, K. Whitehouse, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer, and D. Culler, "TinyOS: An Operating System for Sensor Networks," in *Ambient Intelligence*, 2005, pp. 115–148.

[11] "ISA100.11a: Wireless systems for industrial automation: Process control and related applications," Draft standard, 2009.

[12] "Why WirelessHART?" White Paper, HART Communication Foundation, Oct. 2007.

[13] P. Seiler and R. Sengupta, "Analysis of communication losses in vehicle control problems," in *Proceedings of the American Control Conference*, 2001, pp. 1491–1496.

[14] M. C. de Oliveira, J. E. Camino, and R. E. Skelton, "A convexifying algorithm for the design of structured linear controllers," in *Proceedings of the 39th IEEE Conference on Decision and Control*, 2000, pp. 2781–2786.

[15] P. Naghshtabrizi and J. P. Hespanha, "Anticipative and non-anticipative controller design for network control systems," in *Networked Embedded Sensing and Control*, ser. Lect. Notes in Contr. and Inform. Sci. Springer, 2006, vol. 331, pp. 203–218.

[16] L. El Ghaoui, F. Oustry, and M. Ait Rami, "A cone complementarity linearization algorithm for static output-feedback and related problems," *IEEE Transactions on Automatic Control*, vol. 42, no. 8, pp. 1171–1176, Aug. 1997.

[17] N. Elia, "Remote stabilization over fading channels," *Systems & Control Letters*, vol. 54, no. 3, pp. 237–249, 2005.

[18] C. L. Robinson and P. R. Kumar, "Sending the most recent observation is not optimal in networked control," in *Proceedings of the 46th IEEE Conference on Decision and Control*, 2007, pp. 334–339.